University of Glasgow

DEPARTMENT OF

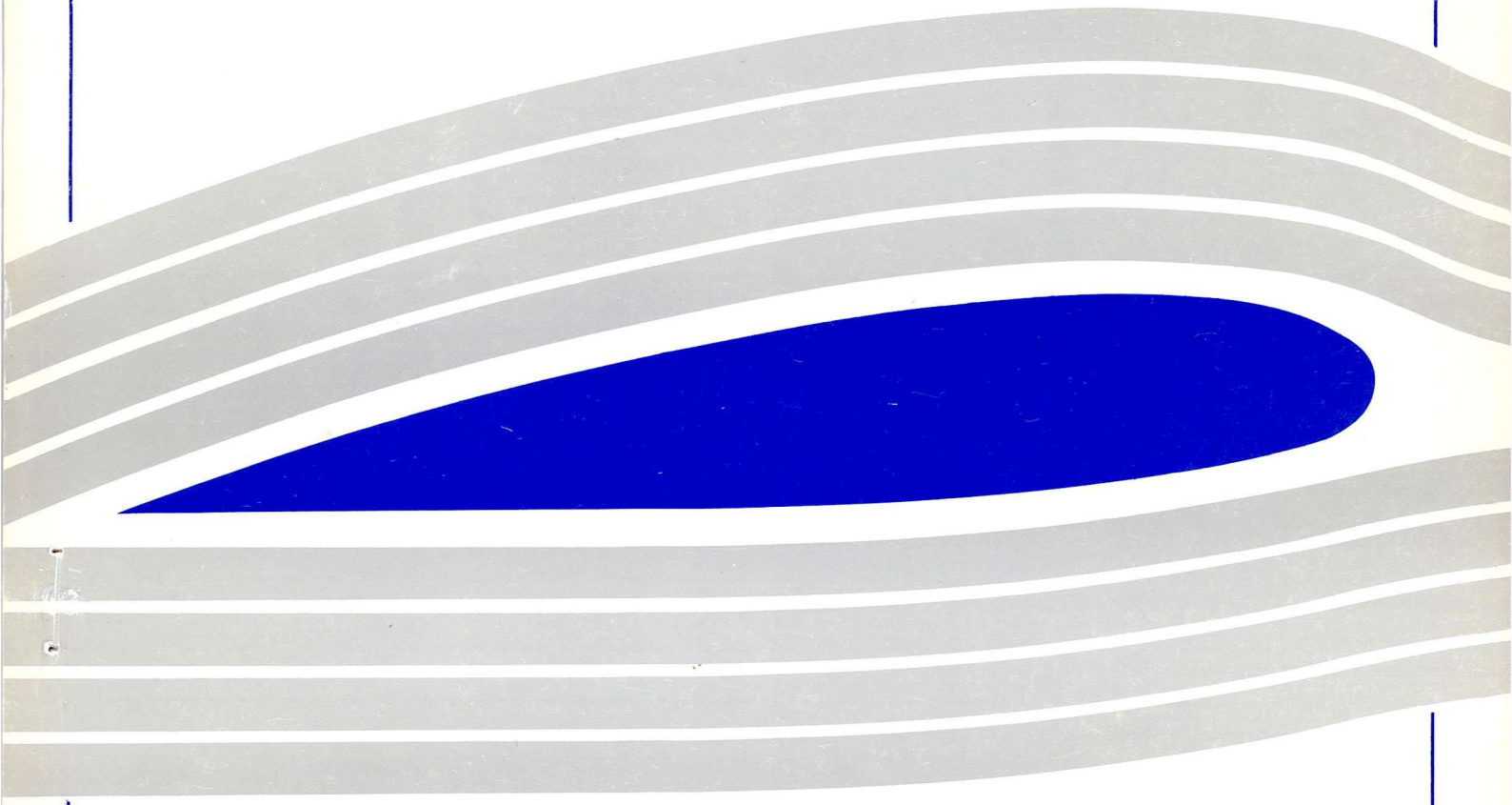
AEROSPACE ENGINEERING

UNIVERSITY OF GLASGOW
- 5 NOV 1996
LIBRARY

Parallel Newton Method for High-Speed Viscous Separated Flowfields

X. Xu, N. Qin and B. E. Richards

Engineering
PERIODICALS



G.U. Aero Report No. 9210

Engineering
PERIODICALS
JS000

Parallel Newton Method for High-Speed Viscous Separated Flowfields

X. Xu, N. Qin and B. E. Richards

Department of Aerospace Engineering
University of Glasgow
Glasgow G12 8QQ

April, 1992.

Parallel Newton Method for High-Speed Viscous Separated Flowfields

X. Xu, N. Qin and B. E. Richards

Department of Aerospace Engineering, University of Glasgow, Glasgow G12 8QQ, U.K.

Abstract

This paper presents a new technique to parallelize Newton method for the locally conical approximate, laminar Navier-Stokes solutions on a distributed memory parallel computer. The method uses Newton's method for nonlinear systems of equations to find steady-state solutions. The parallelization is based on a parallel iterative solver for large sparse non-symmetric linear system. The method of distributed storage of the matrix data results in the corresponding geometric domain decomposition. The large sparse Jacobian matrix is then generated distributively in each subdomain. Since the numerical algorithms on the global domain are unchanged, the convergence and the accuracy of the original sequential scheme are maintained, and no inner boundary condition is needed.

CONTENTS

1. INTRODUCTION

One of the main factors limiting the widespread use of robust and fast convergence Navier-Stokes (N-S) solvers is their requirement for very large computer memory. Recent advances in parallel distributed memory multiprocessors offer the best near-term hope for solving the N-S equations using implicit methods and ultimately Newton's methods. However, how to develop efficient parallel algorithms more suitable to these computer architectures is presently the main task. Recently Venkatakrishnan *et al.* [15], Braaten [2,3], and Tromeur-Dervout *et al.* [13] have done some research in parallel N-S solutions. Radicati *et al.* [11] and Leland *et al.* [5] studied parallel linear system solutions. In this paper, based on the parallel α-GMRES method [16] for solving the large sparse non-symmetric linear system, the authors have developed a new parallel Newton's algorithm for the N-S equations. Using the α-GMRES method [16] Qin *et al.* [8] has presented the sequential Newton's algorithm for the same flowfields as discussion in this paper.

2. GOVERNING EQUATIONS

The locally conical Navier-Stokes equations can be derived through the general coordinate transformation

$$\begin{aligned} x &= x(\xi,\eta,\zeta) = r(\xi)\sin\theta(\eta,\zeta)\cos\varphi(\eta,\zeta) \\ y &= y(\xi,\eta,\zeta) = r(\xi)\sin\theta(\eta,\zeta)\sin\varphi(\eta,\zeta) \\ z &= z(\xi,\eta,\zeta) = r(\xi)\cos\theta(\eta,\zeta) \end{aligned} \tag{2.1}$$

to the 3-D Navier-Stokes equations in Cartesian coordinates (x,y,z). Here $r(\xi)$ is the transformation to the radial coordinate. The parameters $\theta(\eta,\zeta)$ and $\varphi(\eta,\zeta)$ are general two dimensional transformations to fit different conical shapes and control the clustering of grid points. After the locally conical approximation, i.e. the derivatives of flow properties to r are neglected, the following governing equations are obtained [9].

$$\frac{\partial \mathbf{F}}{\partial \eta} + \frac{\partial \mathbf{G}}{\partial \zeta} + \mathbf{H} = 0 \tag{2.2}$$

where

$$\mathbf{Q} = \overline{\mathbf{Q}}/J \ , \quad \overline{\mathbf{Q}} = [\rho, \rho u, \rho v, \rho w, E_t]^T$$

$$\mathbf{F} = \mathbf{F}_i - \mathbf{F}_v \quad , \quad \mathbf{G} = \mathbf{G}_i - \mathbf{G}_v$$

$$\mathbf{F}_i = (\eta_x/J)\,\overline{\mathbf{E}}_i + (\eta_y/J)\,\overline{\mathbf{F}}_i + (\eta_z/J)\,\overline{\mathbf{G}}_i \ , \quad \mathbf{G}_i = (\zeta_x/J)\,\overline{\mathbf{E}}_i + (\zeta_y/J)\,\overline{\mathbf{F}}_i + (\zeta_z/J)\,\overline{\mathbf{G}}_i$$

$$\mathbf{F}_v = (\eta_x/J)\,\overline{\mathbf{E}}_v + (\eta_y/J)\,\overline{\mathbf{F}}_v + (\eta_z/J)\,\overline{\mathbf{G}}_v \ , \quad \mathbf{G}_v = (\zeta_x/J)\,\overline{\mathbf{E}}_v + (\zeta_y/J)\,\overline{\mathbf{F}}_v + (\zeta_z/J)\,\overline{\mathbf{G}}_v$$

$$\overline{\mathbf{E}}_i = [\rho u, \rho u^2 + p, \rho uv, \rho uw, (E_t + p)u]^T$$
$$\overline{\mathbf{F}}_i = [\rho v, \rho uv, \rho v^2 + p, \rho vw, (E_t + p)v]^T$$
$$\overline{\mathbf{G}}_i = [\rho w, \rho uw, \rho vw, \rho w^2 + p, (E_t + p)w]^T$$

$$\overline{\mathbf{E}}_v = [0, \tau_{xx}, \tau_{xy}, \tau_{xz}, u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x]^T$$
$$\overline{\mathbf{F}}_v = [0, \tau_{xy}, \tau_{yy}, \tau_{yz}, u\tau_{xy} + v\tau_{yy} + w\tau_{yz} - q_y]^T$$
$$\overline{\mathbf{G}}_v = [0, \tau_{xz}, \tau_{yz}, \tau_{zz}, u\tau_{xz} + v\tau_{yz} + w\tau_{zz} - q_z]^T$$

$$\mathbf{H} = (2\overline{\mathbf{E}}_i - \overline{\mathbf{E}}_v)/\xi$$

3. DISCRETIZATION AND NEWTON METHOD

In order to capture both the strong shock waves and the shear layers for accurate prediction of the flowfield and, therefore, accurate prediction of the heat transfer rate distributions, the choice of numerical schemes becomes an important factor in the computer simulation.

3.1. High Resolution Scheme Formulation

In the cell centred finite volume formulation employed here, the state variables $\mathbf{Q}$ are evaluated at cell centres and represent cell-averaged values. The fluxes $\mathbf{F}_i$ and $\mathbf{G}_i$ are evaluated at cell interfaces. The spatial derivatives are then represented as a flux balance across a cell. The interface flux is determined from a local one-dimensional model of wave interactions normal to the cell interfaces. With the flux difference splitting (FDS) model developed by Osher *et al* [6,7], the interface flux can be written as

$$\tilde{\mathbf{F}}_i = 1/2 \left[\mathbf{F}_i(\mathbf{Q}^L) + \mathbf{F}_i(\mathbf{Q}^R) - \int_{Q^L}^{Q^R} \left| \frac{\partial \mathbf{F}_i}{\partial \mathbf{Q}} \right| d\mathbf{Q} \right] \tag{3.1}$$

3.2. The MUSCL Scheme for High Order Accuracy and the Limiter

The state-variable interpolations determine the resulting accuracy of the scheme. A k-parameter family of higher-order schemes [14,1] can be written as

$$\mathbf{Q}^{L}_{i+1/2,j} = \mathbf{Q}_{i,j} + \left\{\left(\frac{s}{4}\right)\left[(1-ks)\Delta_{-} + (1+ks)\Delta_{+}\right]\right\}_{i,j}$$
$$\mathbf{Q}^{R}_{i+1/2,j} = \mathbf{Q}_{i+1,j} - \left\{\left(\frac{s}{4}\right)\left[(1-ks)\Delta_{+} + (1+ks)\Delta_{-}\right]\right\}_{i+1,j} \qquad (3.2)$$

where

$$s = \frac{2\Delta_{+}\Delta_{-} + \varepsilon}{(\Delta_{+})^{2} + (\Delta_{-})^{2} + \varepsilon}$$
$$(\Delta_{+})_{i,j} = \mathbf{Q}_{i+1,j} - \mathbf{Q}_{i,j}, \qquad (\Delta_{-})_{i,j} = \mathbf{Q}_{i,j} - \mathbf{Q}_{i-1,j}$$

3.3. The Newton Method

The basic solution procedure is known as Newton's method for systems of nonlinear equations. Systems of this type have the form

$$\mathcal{F}(\mathbf{Q}) = 0 \qquad (3.3)$$

The general Newton's method is

$$\left(\frac{\partial \mathcal{F}}{\partial \mathbf{Q}}\right)^{n} \Delta^{n}\mathbf{Q} = -\mathcal{F}^{n}(\mathbf{Q}) \qquad (3.4)$$

By forming $\mathcal{F}(\mathbf{Q})$ and the Jacobian $\partial\mathcal{F}/\partial\mathbf{Q}$ at the known nth iterate, the increment $\Delta^{n}\mathbf{Q}$ is then found by solving the linear system. The value of $\mathbf{Q}$ at the new iterate is given by

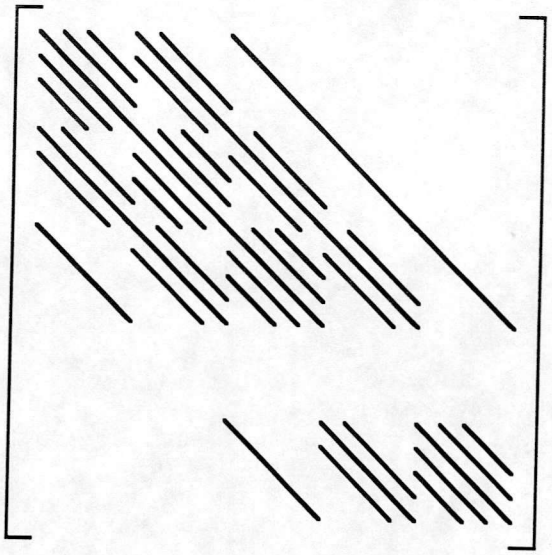
$$\mathbf{Q}^{n+1} = \mathbf{Q}^{n} + \Delta^{n}\mathbf{Q} \qquad (3.5)$$

4. PARALLEL α-GMRES METHOD

We denote the linear system by

$$\mathcal{A}x = b \qquad (4.1)$$

where the structure of $\mathcal{A}$ depends on the spatial discretization scheme used. Typically we consider the following system resulting from a second or third order high resolution scheme using a structured grid for a two-dimensional Navier-Stokes solution. The linear system will be a block 13-point diagonal matrix which can be denoted as



4.1. The α-GMRES Method

For the large sparse non-symmetric linear system $\mathcal{A}x = b$, the α-GMRES method [16] is written as

$$(\alpha I + \mathcal{D}^{-1}\mathcal{A})\, x^{n+1} = \mathcal{D}^{-1}b + \alpha x^{n}. \qquad (4.2)$$

Given x^n, the above equation is solved for x^{n+1} using the GMRES method [12]. This procedure is continued until the sequence x^n is converged, and the convergent vector is the solution of the original linear system. This procedure could be thought as an inner iterative loop of GMRES algorithm combined with the outer iterative loop of α-GMRES algorithm. Here $\mathcal{D}$ is a block diagonal matrix of $\mathcal{A}$, I is a unit matrix, and $\alpha > 0$.

4.2. The Parallel α-GMRES Method

It is assumed there are P processors available.

4.2.1. Data Distribution

The matrix $\mathcal{A}$ can be written in columns as $\mathcal{A} = [\, \mathcal{A}^1, \mathcal{A}^2, \ldots, \mathcal{A}^P \,]$, where $\mathcal{A}^p$ are N×L submatrics, p=1,2,...,P. A vector v can be written as

$$v = \begin{pmatrix} v^1 \\ v^2 \\ \vdots \\ v^P \end{pmatrix}$$

where v^p is vector of order L corresponding to $\mathcal{A}^p$, p=1,2,...,P. $\mathcal{A}^p$ and v^p are stored in each processor p. The distribution of the matrix data in columns does not increase the data storage compared to the sequential case.

Remark: The L in the order of N×L may be not the same in different processors.

4.2.2. Parallel Algorithm

Let ε_1 be the convergence criterion of the inner GMRES algorithm and ε_2 be the convergence criterion of the outer loop of the α-GMRES algorithm. In processor p, we perform the following calculations and communications.

Step 1: Initialization

Set an initial guess $x^p{}_0$, we have

$$\mathcal{A}^p x^p{}_0 = \bar{r}^p{}_0 , \qquad (*)$$

$$r^p = b^p - \bar{r}^p ,$$

and

$$\|r_0\| = \sqrt{\sum_{p=1}^{P} (r^p{}_0 , r^p{}_0)} \quad . \qquad (**)$$

Let $\delta = \|r_0\|$ and $w^p{}_0 = x^p{}_0$.

Remarks:

(*): Matrix-vector multiplication can be generated as follows:

$$\mathcal{A} v_i = (\mathcal{A}^1, \mathcal{A}^2, \cdots, \mathcal{A}^P)\left(\begin{pmatrix} v^1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}_i + \begin{pmatrix} 0 \\ v^2 \\ \vdots \\ 0 \end{pmatrix}_i + \cdots + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ v^P \end{pmatrix}_i\right) = \mathcal{A}^1 v^1{}_i + \mathcal{A}^2 v^2{}_i + \cdots + \mathcal{A}^P v^P$$

$$= \begin{pmatrix} * \\ * \\ \vdots \\ * \end{pmatrix} + \begin{pmatrix} * \\ * \\ \vdots \\ * \end{pmatrix} + \cdots + \begin{pmatrix} * \\ * \\ \vdots \\ * \end{pmatrix} \Rightarrow \begin{pmatrix} \bar{v}^1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}_i + \begin{pmatrix} 0 \\ \bar{v}^2 \\ \vdots \\ 0 \end{pmatrix}_i + \cdots + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \bar{v}^P \end{pmatrix}_i$$

where "$\Rightarrow$" indicates the communication of data among different processors to form $\bar{v}_i$. In this way, the task of calculating $\mathcal{A} v_i$ for P processors is divided by calculating $\mathcal{A}^p v^p_i$ on processor p. This is the main calculation in the GMRES method. The resulting vector $\bar{v}_i$ is again distributed to the P processors. The only communication required in the calculation is in the formation of $\bar{v}_i$. Due to the sparsity of the matrix $\mathcal{A}$, this communication is only of a limited nature.

(**): Here requires the collection of the partial inner products carried out on each processor.

Step 2: Calculate $\mathcal{B} = (\alpha I + \mathcal{D}^{-1}\mathcal{A})$ and $c = \mathcal{D}^{-1} b$

We can write $\mathcal{B}$ in columns as $\mathcal{B} = [\ \mathcal{B}^1, \mathcal{B}^2, ..., \mathcal{B}^P\ ]$, which has the same stencil as matrix $\mathcal{A}$. Parts of $\mathcal{D}^{-1}$ are calculated in each processor separately. So $\mathcal{D}^{-1}\mathcal{A}$ can be performed in each processor provided that appropriate communications are arranged. Then the α is added in diagonal elements in each processor so that we have $\mathcal{B}^p$ in each processor. $c = \mathcal{D}^{-1}b$ can be performed in each processor without any communication.

Step 3: Calculation

Let $e^p_0 = c^p + \alpha\, w^p_0$ and we have

$$\mathcal{B}^p w^p_0 = \bar{f}^p_0 ,$$

$$f^p_0 = e^p_0 - \bar{f}^p_0 ,$$

and then set

$$\hat{v}^p_1 = f^p_0 ,$$

so we have

$$\|f_0\| = \sqrt{\sum_{p=1}^{P} (f^p_0 , f^p_0)} \quad .$$

$$v^p_1 = \frac{f^p_0}{\|f_0\|} \quad .$$

Let $\delta_1 = \|f_0\|$ in the first iteration.

Step 4: For i=1 to k

$$\mathcal{B}^p v^p_i = \bar{v}^p_i ,$$

the elements of the Hessenberg matrix are calculated using

$$\beta_{i+1,j} = \sum_{p=1}^{P} (\bar{v}^p_i , v^p_j) \quad .$$

We then calculate

$$\hat{v}^p_{i+1} = \bar{v}^p_i - \sum_{j=1}^{i} \beta_{i+1,j} v^p_j ,$$

and

$$\|\hat{v}_{i+1}\| = \sqrt{\sum_{p=1}^{P} (\hat{v}^p_{i+1} , \hat{v}^p_{i+1})} \quad ,$$

and normalise the base vector as follows

$$v^p_{i+1} = \frac{\hat{v}^p_{i+1}}{\|\hat{v}_{i+1}\|} \quad .$$

After k steps, the Hessenberg matrix is

$$\mathcal{H}_k = \begin{pmatrix} \beta_{2,1} & \beta_{3,1} & \cdots & \beta_{k+1,1} \\ \|\hat{v}_2\| & \beta_{3,2} & \cdots & \beta_{k+1,2} \\ 0 & \|\hat{v}_3\| & \ddots & \vdots \\ \vdots & \vdots & \ddots & \beta_{k+1,k} \\ 0 & 0 & \cdots & \|\hat{v}_{k+1}\| \end{pmatrix}_{(k+1)\times k} .$$

Step 5: Uses a Q-R algorithm to find y such that

$$\| \delta_2 e_1 - \mathcal{H}_k y \| = \min_{y_0 \in R^k} \| \delta_2 e_1 - \mathcal{H}_k y_0 \| \quad ,$$

where $y = (y_1, y_2,...,y_k)^T$, $e_1 = (1,0_1,...,0_k)^T$ and $\delta_2 = \|f_0\|$, so we have $w^p = w^p{}_0 + \sum y_k v^p{}_k$.

Step 6: Calculation

$$\mathcal{B}^p w^p = \bar{f}^p \quad ,$$

$$f^p = e^p{}_0 - \bar{f}^p \quad ,$$

and

$$\|f\| = \sqrt{\sum_{p=1}^{P} (f^p, f^p)} \quad .$$

If $\|f\| < \delta_1 \times \varepsilon_1$ then we go to next step, or else let $w^p{}_0 = w^p$ and go to step 3.

Step 7: Calculation

$$\mathcal{A}^p w^p = \bar{r}^p \quad ,$$

$$r^p = b^p - \bar{r}^p \quad ,$$

and

$$\|r\| = \sqrt{\sum_{p=1}^{P} (r^p, r^p)} \quad .$$

If $\|r\| < \delta \times \varepsilon_2$ then the algorithm is stopped, else we let $w^p{}_0 = w^p$ and go to step 3.

5. PARALLEL GENERATION OF THE SPARSE JACOBIAN MATRIX

Consider the solution of the nonlinear system given by Eq. (3.3) using Newton's method. The approach is to replace the analytic Jacobian with a numerically approximated Jacobian. Qin and Richards [10] have successfully applied the method for the estimation of sparse Jacobian matrices [4] for the Navier-Stokes solutions. In the sequential computation

case: Let h be square root of machine epsilon, if $\mathcal{J}$ is a band matrix of band-width m=2ρ-1 then the difference $\mathcal{F}_i(\mathbf{Q}+h\ \mathbf{e}_j) - \mathcal{F}_i(\mathbf{Q})$ is zero if $|\ i - j\ | \geq \rho$; it follows that we may find simultaneously approximations to columns j+km, k = 0,1,2,..., of $\mathcal{J}$ from the difference $\mathcal{F}_i(\mathbf{Q}+\Sigma\ h\ \mathbf{e}_{j+km}) - \mathcal{F}_i(\mathbf{Q})$. Here the sum Σ is for the subscript k. In this way the total number of subroutine calls needed can be reduced from n+1 to m+1. This strategy positively minimizes the total number of function evaluations in view of the number of unknown coefficients in each row of $\mathcal{J}$. In the parallel case: Because the property of forming the Jacobian is according to columns the Jacobian $\mathcal{J}$ can be generated in each subdomain. Here this subdomain compared with the subdomain corresponding to the matrix storage case is relative large, i.e. it is based on the original subdomain with each one extended 4 grid points into the neighbouring subdomain.

6. NUMERICAL TESTS AND DISCUSSION

The foregoing numerical tests have been carried out on the test case of a laminar Mach 7.95 flow around a sharp cone with a cold wall (T_w =309.8K) and at an angle of attack of 24°. The Reynolds number is 4.1×10^6 and the flow temperature is 55.4K. This case produces a flow which has a large separated flow region with embedded shock waves on the leeward side of the cone and strong gradients in the thin boundary layer on the windward side. Accurate validation with experiment was achieved in the flow field and heat transfer distribution. The spatial discretization scheme used is the Osher flux difference splitting scheme. The formal accuracy is third order for the convective fluxes and second order for the diffusive fluxes.

The grids in the cross section tested in the two cases are 34×34, and 66×34. The method of distributed storage of the matrix data results in the corresponding geometric domain decomposition which is illustrated in Fig.1. The resulting large sparse non-symmetric linear systems to be solved are block 13-point structured matrices of order 32×32×5 and 64×32×5 corresponding to the different grids.

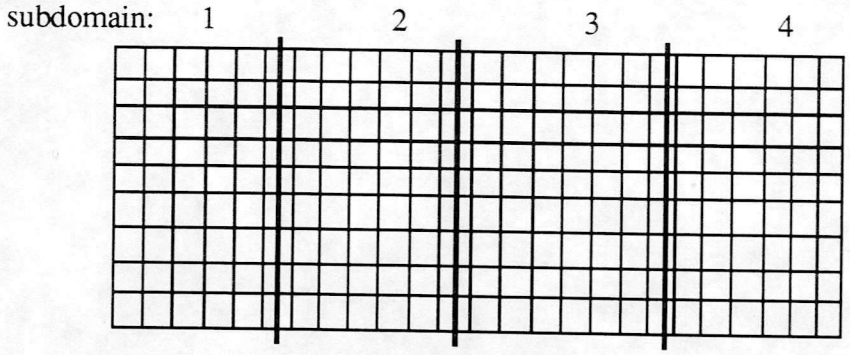


Fig. 1: The domain decomposition

The parallel algorithm has been tested on a distributed memory computer, the University of Glasgow Meiko Computing Surface composed of T800 transputers. Fig.2 shows the speedup achieved using from 1 to 8 processors for solving the linear system using the α-GMRES algorithm. Fig.3 shows the convergence histories for different numbers of processors. The convergence criterion of the inner GMRES algorithm ε_1 is 10^{-1} and the convergence criterion of the outer loop of the α-GMRES algorithm ε_2 is 10^{-10}. Fig.4 shows the speedup achieved using from 1 to 8 processors for solving the Navier-Stokes equation using the Newton's method. Fig.5 shows the convergence histories for different numbers of processors, the convergence criterion of the inner GMRES algorithm ε_1 is 10^{-1}, the convergence criterion of the outer loop of α-GMRES algorithm ε_2 is 10^{-2}, and the convergence criterion of the whole Navier-Stokes solution ε_3 is 10^{-10}. As for the classical Newton method, a good initial guess is an important aspect. The initial guess used here was provided by an explicit time dependent approach using the Runge-Kutta method with local time stepping, which is robust when starting the solution from free stream conditions but slow in convergence. Fig.6 shows the memory required on each processor for solving the Navier-Stokes equation. As can be seen, the requirement on the memory for each processor decreases as the number of the processors increases. To give an impression of the flowfield solved, fig.7 shows the temperature contours of the solved flow field.

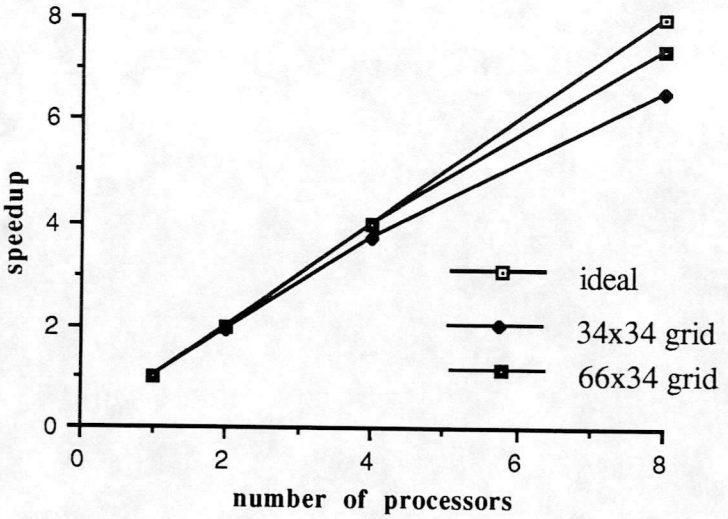


Fig. 2: Speedup with different grids for α-GMRES algorithm

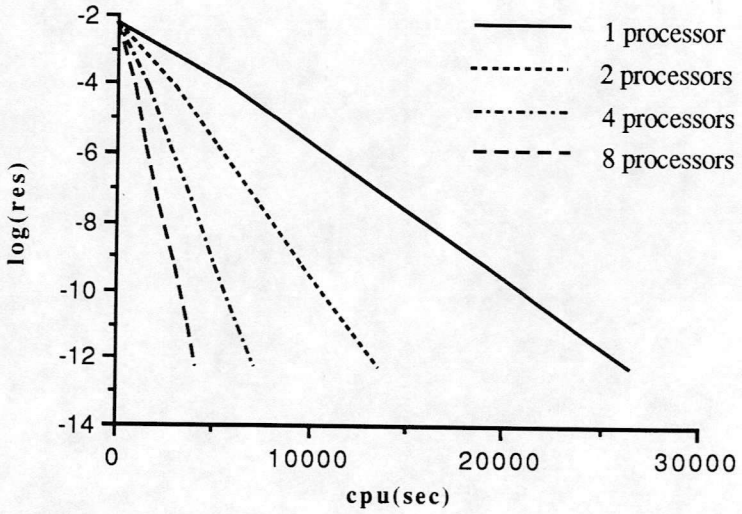


Fig. 3: Convergence of α-GMRES algorithm with different number of processors

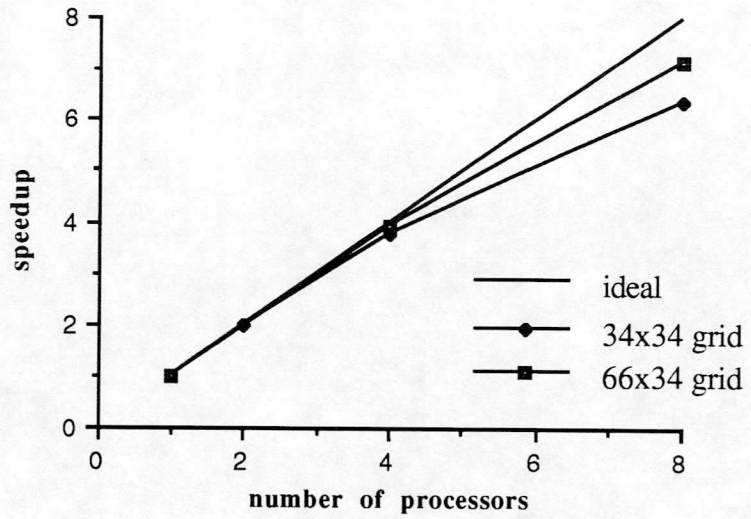


Fig. 4: Speedup with different grids for N-S solution

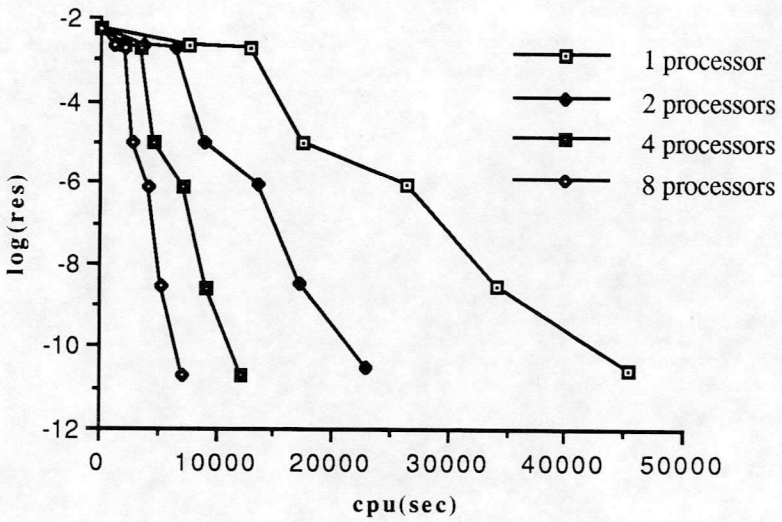


Fig. 5: Convergence of N-S solution with different number of processors

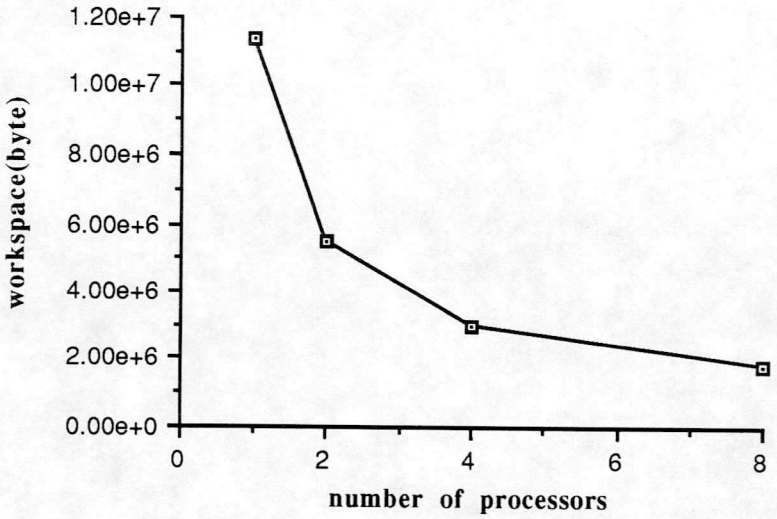


Fig. 6: Memory needed for N-S solution with different number of processors in 66×34 grid case

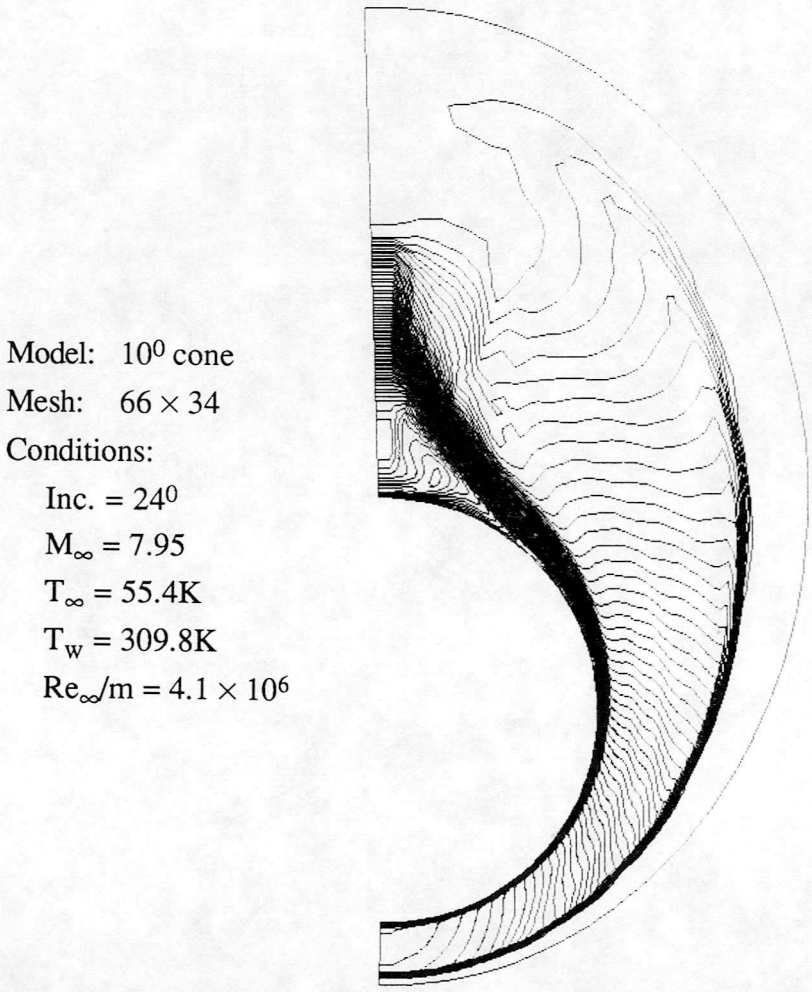


Fig. 7: Temperature contours of cross sectional view of the cone flowfield

7. CONCLUDING REMARKS

An efficient and fast convergent parallel laminar Navier-Stokes solver with a high resolution flux difference split scheme has been developed for hypersonic compressible flow around conical shape on a distributed memory parallel computer. The flowfield simulated include both strong shock waves and a large separated region. The parallelization is based on the parallel α-GMRES solver for the large sparse non-symmetric linear system. The parallel scheme maintains the convergence and the accuracy of the original sequential scheme and does not increase any inner boundary condition. In the parallel scheme we store the elements of Jacobian of the linear system with no overlap in different processors, i.e. the sum of the storage of the matrix elements in each processor is the same as the storage of the matrix elements in the sequential scheme, which is the main storage of the Newton's method. This data storage method provides positive scope for solving the full 3-D Reynold's averaged Navier-Stokes equation using Newton's method in the future.

ACKNOWLEDGMENTS

The authors thank Dr. Derek Higgins of the computer centre of University of Glasgow for his help in using the Meiko Computing Surface. The first author wishes to thank the University of Glasgow for a research scholarship and the Secretary of State for Education and Science for ORS award.

REFERENCES

1. Anderson, W.K., Thomas, J.L., and Van Leer, B. A comparison of finite volume flux vector splittings for the Euler equations. *AIAA J.* Vol. 24, (9), 1986, pp. 1453-1460.
2. Braaten, M.E. Solution of viscous fluid flows on a distributed memory concurrent computer. *Int. J. Numer. Methods Fluids*, Vol. 10, 1990, pp. 889-905.
3. Braaten, M.E. Development of a parallel computational fluid dynamics algorithm on a hypercube computer. *Int. J. Numer. Methods Fluids*, Vol. 12, 1991,

pp. 947-963.

4. Curtis, A., Powell, M.J.D and Reid, J.K. On the Estimation of Sparse Jacobian Matrices. *J. Inst. Maths. Applics.* Vol. 13, 1974, pp. 117-119.
5. Leland, R.W. and Rollett, J.S. Parallel extrapolation methods for computational fluid dynamics. *Lect. Notes in Phys.* Vol. 371, 1990, pp. 308-312.
6. Osher, S. and Solomon, F. Upwind schemes for hyperbolic systems of conservation laws. *Math. Comp.* Vol. 38, 1982, pp. 339-374.
7. Osher, S. and Chakravarthy, S.R. Upwind schemes and boundary conditions with applications to Euler equations in general coordinates. *J. Comp. Phys.* Vol. 50, 1983, pp. 447-481.
8. Qin, N., Xu, X., and Richards, B.E. SFDN–α-GMRES and SQN–α-GMRES methods for fast high resolution NS simulations. To appear in *Proc. Conference on Numerical Methods for Fluid Dynamics*. Reading 7-10, April 1992.
9. Qin, N., Scriba, K.W., and Richards, B.E. Shock-shock, shock-vortex interaction and aerodynamic heating in hypersonic corner flow. Aeronautical J. Vol. 95, No. 945, 1991, pp. 152-160.
10. Qin, N. and Richards B. E. Sparse Quasi-Newton method for Navier-Stokes Solutions. *Notes in Numerical Fluid Mechanics*, Vol. 29, 1990, pp. 474-483.
11. Radicati, G. and Robert, Y. Parallel conjugate gradient-like algorithms for solving sparse nonsymmetric linear systems on a vector multiprocessor. *Parallel Computing*, Vol. 11, 1989, pp. 223-239.
12. Saad, Y and Schultz, M.H. GMRES: A general Minimal Residual Algorithm for Solving Nonsymmetric Linear System. *SIAM J. Stat. Comp.*, Vol. 7, No. 3, 1986, pp. 856-869.
13. Tromeur-Dervout, D., Phuoc, L.TA., and Mane, L. A 3-D Navier-Stokes solver on distributed memory multiprocessor. *Lect. Notes in Phys.* Vol. 371, 1990, pp. 303-307.
14. Van Leer, B. Upwind-difference methods for aerodynamic problems governed by the Euler equations. *Lect. Appl. Math.* Vol. 22, Part II, 1985, pp. 327-336.

15. Venkatakrishnan, V., Saltz, J.H., and Mavriplis, D.J. Parallel preconditioned iterative methods for the compressible Navier-Stokes equations. *Lect. Notes in Phys*. Vol. 371, 1990, pp. 233-237.

16. Xu, X., Qin, N., and Richards, B.E. α-GMRES: A new parallelizable iterative solver for large sparse non-symmetric linear system arising from CFD. *G.U. Aero Report*, No.9110, 1991. To appear in *Int. J. Numer. Methods Fluids*.